COSC344
Database Theory and Applications

Lecture 1: Introduction

Welcome to

COSC 344

Database Theory and Applications

Course Goals

- Introduce the fundamental concepts, principles, and problems in database.
- Introduce the principles of **database design, normalization** and **implementation**
- Learn the **design** and **implementation** of the underlying database management system software

What will you get from this course?

↓

Necessary **knowledge** and **fundamental skills** you'll need in your future career

Theory + Practice

Teaching Team

- Lecturers

  Haibo Zhang (Course Coordinator)
  2.47, Owheo
  479-8534
  haibo@cs.otago.ac.nz

  Yawen Chen
  2.46, Owheo
  479-5740
  yawen@cs.otago.ac.nz

- Lab demonstrator

  Pradeesh Parameswaran
  prapr694@student.otago.ac.nz

Course Details

- Recommended Textbook
  - Elmasri, R. & Navathe, S., Fundamentals of Database Systems, (7th edition), 2016
- Teaching materials
  - Course webpage: http://www.cs.otago.ac.nz/cosc344
  - Blackboard: lecture recording, assignment submission
- Communication channel
  - use Blackboard to communicate about the course
  - Emails will be sent to your student.otago.ac.nz account

    Check your university email account regularly!
- Almost one lab per week (no lab in week 1, 10, and 13)
  - Two streams, only need to attend one

Outline of Lectures

- Introduction to databases, models
- Database design, ER modeling
- Relational model
- Relational algebra & operator
- Functional dependencies
- Normalization
- View & NULL

Database fundamentals and design

- SQL, Java & SQL,C & SQL, PHP &SQL
- PL/SQL, Triggers

Database programming

- DBMS Architecture & System catalog
- Database file and storage
- Database indexing (1 & 2)
- Database security & auditing
- Transactions
- Concurrency control
- Query optimisation

DBMS architecture, design and optimization

- NoSQL and big data

Non-relational models

Assessment

- Two assignments - 25%
  - Assignment 1 (10%, due on March 29)
  - Assignment 2 (15%, due on May 17)
  - Late submission policy: 10% penalty per working day late
  - Group assignment: 4 per group, randomly created
  - Submit through Blackboard
- Lab assessments -15%
  - Not every lab has assessment – check labnotes
  - Submit through Blackboard
- Exam - 60%
  - 3 hours
  - A mark of **at least 50%** is required to pass the course

Academic Integrity and Academic Misconduct

Academic integrity means being honest in your studying and assessments. It is the basis for ethical decision-making and behaviour in an academic context. Academic integrity is informed by the values of honesty, trust, responsibility, fairness, respect and courage. Students are expected to be aware of, and act in accordance with, the University's Academic Integrity Policy.

Academic Misconduct, such as plagiarism or cheating, is a breach of Academic Integrity and is taken very seriously by the University. Types of misconduct include plagiarism, copying, unauthorised collaboration, taking unauthorised material into a test or exam, impersonation, and assisting someone else's misconduct. A more extensive list of the types of academic misconduct and associated processes and penalties is available in the University's Student Academic Misconduct Procedures.

Academic Integrity Continued

It is **your responsibility** to be aware of and use acceptable academic practices when completing your assessments. To access the information in the Academic Integrity Policy and learn more, please visit the Universitys Academic Integrity website at www.otago.ac.nz/study/academicintegrity or ask at the Student Learning Centre or Library.

- Academic Integrity Policy
  www.otago.ac.nz/administration/policies/otago116838.html

- Student Academic Misconduct Procedures
  http://www.otago.ac.nz/administration/policies/otago116850.html

Learning Objectives of Lecture 1

- You should
  - Understand **database** and **database management system**
  - Be able to explain the advantage of managing data using the database approach
  - Understand **data modelling** fundamentals such as data model, database schema, and three-schema architecture
  - Understand **data independence** and its advantage

- Source
  - Textbook: Chapters 1, Chapter 2.1 – 2.4

Try It

- List some ways you encounter databases in everyday activities.

A Motivating Example - eVision

- To store the information about:
  - students
  - courses
  - staff
  - who takes what, who teaches what
- Allow users to query/update:
  - who teaches “COSC344”, enroll “Mary” in “COSC344”
- Allow >1000 users to access data simultaneously
- Store the data for a long period of time
  - Protect against crashes
  - Protect against unauthorized use

Introduction to Databases (1)

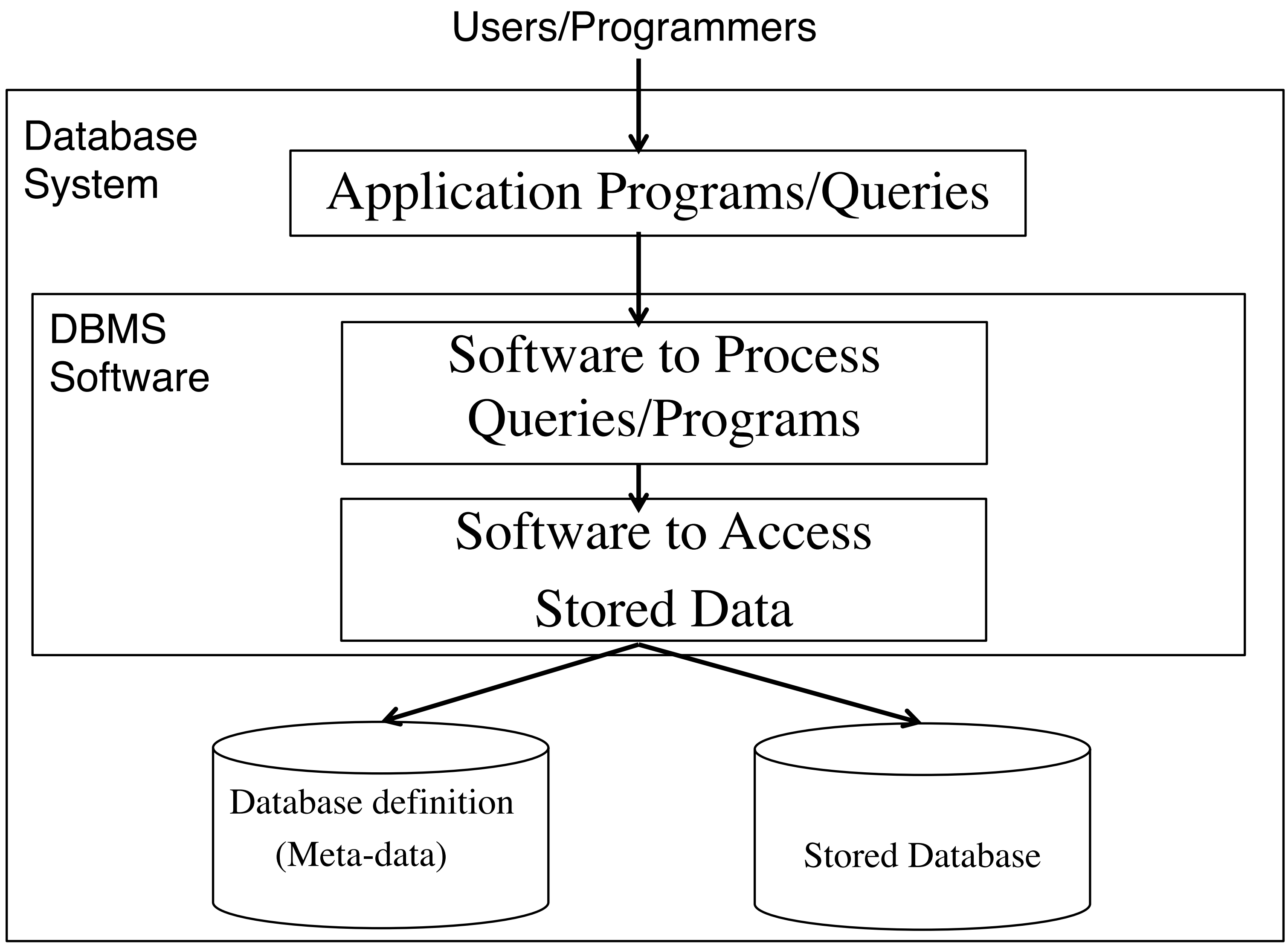
- What is a database?
  - A database is a **collection** of **related** data.
- A database generally has the following implicit properties:
  - Represents some aspect of the real-word, called **miniworld**
  - Is a coherent collection of data with **inherent meaning**
  - Is designed, built, and populated with data for **a specific purpose**.
- A database can be of any size and complexity
  - Small database: Address book, student database in CS
  - Large database: Inland Revenue Department database,
    Amazon.com,

Large Databases

Introduction to Databases (2)

- What is a database management system (DBMS)?
  - A **collection of programs** that enable to **define, construct, manipulate, and share databases** among various users and applications.
  - Commercial DBMSs: DB2, Oracle, MS SQL Server …
  - Open source: MySQL, PostgreSQL, MariaDB, SQLite
- Application program
  - Accesses the database by sending queries and requests for data to the DBMS

A Simplified Database System Environment



An Example Database

STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

COURSE	CourseName	CourseNumber	Credits	Dept
	Intro to Comp Sci	COSC1310	4	CS
	Data Structures	COSC3320	4	CS
	Discrete Maths	MATH2410	3	MATH
	Database	COSC3380	3	CS

SECTION	SectionID	CourseNumber	Semester	Year	Instructor
	85	MATH2410	Fall	98	King
	92	COSC1310	Fall	98	Anders
	102	COSC3320	Spring	99	Knuth
	112	MATH2410	Fall	99	Chang
	119	COSC1310	Fall	99	Anders
	135	COSC3380	Fall	99	Turing

GRADE_REPORT	StudentNumber	SectionID	Grade
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PREREQUISITE	CourseNumber	PrerequisiteNumber
	COSC3380	COSC3320
	COSC3380	MATH2410
	COSC3320	COSC1310

Example database (Adapted from Elmasri & Navathe Fig 1.2)

Characteristics of the Database Approach

- Self-describing nature of a database system
  - Database + Meta-data (a complete definition of data structure and constraints stored in DBMS catalog)
- Insulation between programs and data
  - Program-data independence
  - Program-operation independence

  data abstraction
- Support of multiple views of data
  - A subset of the database
  - Virtual data derived from the database files
- Sharing of data
  - Concurrency control
  - Online transactions processing

Actors on the Scene

- Database Administrators (DBA)
  - Authorizing access to the database
  - Coordinating and monitoring its use
  - Maintain software and hardware resources
- Database Designers
  - Identify the data to be stored and choose the appropriate structure to represent and store the data
- End Users
- System Analysts and Application Programmers.

Data Modeling and Data Models

- Data modeling: the first step in designing a database
  - the process of creating a specific data model for a determined problem domain
- A data model is a collection of concepts that can be used to describe **the structure of a database** such as data types, relationships, and constraints that should hold for the data.

Categories of Data Models

- High-level or **conceptual** data models
  - close to the way most users perceive data
  - entity
  - attribute
  - relationship
- Low-level or **physical** data models
  - storage details
- **Representational** or **implementation** data models
  - bridge the gap
  - relational, hierarchical, network
  - NoSQL: key-value, document, graph

Database Schema

- **Database Schema:** the description of a database

STUDENT	Name	StudentNumber	Class	Major

COURSE	CourseName	CourseNumber	Credits	Dept

SECTION	SectionID	CourseNumber	Semester	Year	Instructor

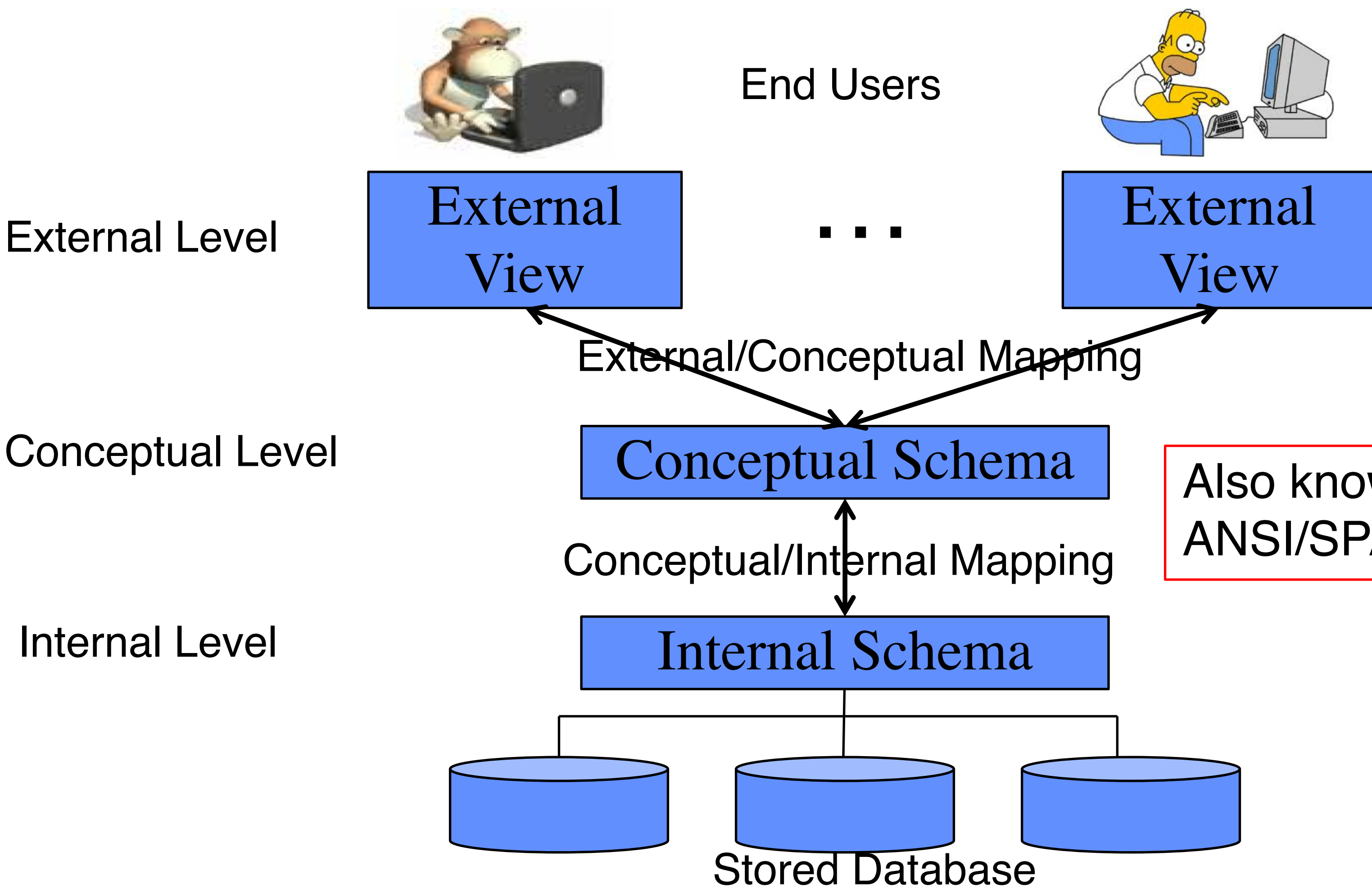
GRADE_REPORT	StudentNumber	SectionID	Grade

PREREQUISITE	CourseNumber	PrerequisiteNumber

Example Database Schema (Adapted from Elmasri & Navathe Fig 2.1)

Three-Schema Architecture (1)

- **Goal:** to separate the user applications from the physical database



Also known as the ANSI/SPARC architecture

Three-Schema Architecture (2)

- Internal Schema
  - Uses a physical data model
  - Describes the physical storage structure of the database
- Conceptual Schema
  - Hides the details of the physical storage structures
  - Concentrates on describing entities, data types, relationships, user operations, and constraints
- External Schema
  - Describes the part of the database that a particular user group is interested in
  - Hides the rest of the database from that user group
- Mapping
  - Transform requests and results between adjacent levels

Data Independence

Defined as **the ability to change the schema at one level of a database system without having to change the schema at the next higher level**

- **Logical** data independence
  - The capacity to change the conceptual schema without having to change external schemas
  - e.g. add/remove record type, change constraints
- **Physical** data independence
  - The capacity to change the internal schema without having to change the conceptual schemas
  - e.g. creating additional access structures

DBMS Languages

- Data definition language (DDL)
  - Used to define the conceptual schema
  - A DDL compiler is used to process DDL statements
- Storage definition language (SDL)
  - Used to define the internal schema
  - Most relational DBMS do not have specific SDL languages
- View definition language (VDL)
  - To specify user views and their mapping to conceptual schema
  - Most DBMSs use DDL for both conceptual and external schema
- Data manipulation language (DML)
  - High-level or nonprocedural DML (set-at-a-time)
  - Low-level or procedural DML (record-at-a-time)

Question to Ponder

- Given some mini-world, how do we design a database?
  - What do you put into it?
  - How are the pieces interrelated?